Assisted Policy Management for SPARQL Endpoints Access Control

Luca Costabello, Serena Villata*, Iacopo Vagliano, and Fabien Gandon

INRIA Sophia Antipolis, France
{firstname.lastname}@inria.fr

Abstract. Shi3ld is a context-aware authorization framework for protecting SPARQL endpoints. It assumes the definition of access policies using RDF and SPARQL, and the specification of named graphs to identify the protected resources. These assumptions lead to the incapability for users who are not familiar with such languages and technologies to use the authorization framework. In this paper, we present a graphical user interface to support dataset administrators to define access policies and the target elements protected by such policies.

1 Introduction

Shi3ld[1] [2] is an access control framework for querying Web of Data servers. It protects RDF stores from incoming SPARQL queries, whose scope is restricted to triples included in accessible named graphs only [1]. In particular, Shi3ld determines the list of accessible graphs by evaluating pre-defined access policies against client attributes sent with the query. It adopts exclusively Semantic Web languages, reuses existing proposals, and protects data up to triple level. The drawback of such framework is that it relies on the assumption that dataset administrators have a proficient knowledge of RDF and SPARQL, and that they are able to manage vocabularies and define new named graphs. In this paper, we address this open issue by presenting a web application that allows non-expert dataset administrators to manage Shi3ld context-aware access control policies, by hiding the complexity of RDF and SPARQL. The Shi3ld policy manager allows the definition of context-aware access conditions featuring user, environment (time and location above all), and device attributes. Moreover, such application allows a simpler definition of new named graphs over a set of existing triples. The work presented in this paper can be classified among the works trying to hide the complexity of SPARQL and the Semantic Web to end users [4–6]. Such proposals mainly consist in GUIs to query, search, visualize, browse and edit triples published on the Web of Data. In our work, we deal with querying issues and we tackle the problem of providing a user-friendly interface for the creation of context-aware access control policies for triple stores.

* The author acknowledges support of the DataLift Project ANR-10-CORD-09 founded by the French National Research Agency.

[1] http://wimmics.inria.fr/projects/shi3ld/

2 Our Proposal

The Shi3ld policy management GUI[2] is designed to support the interaction with two kinds of dataset administrators: *non-experts*, which are assumed not to know the SPARQL query language and RDF, and *experts*, which are able to edit access policies source code. In particular, the following functionalities are proposed:

- **Policies visualization and modification**: the application shows the list of policies stored in the triple store through a grid view. Each policy is an expandable row that, if selected, shows the main features of the policy like the policy target (i.e. the named graphs protected by the policy), the privilege granted by the policy (Create, Update, Read, Delete), and the access conditions (SPARQL 1.1 `ASK` queries) which specify the requirements that need to be satisfied to access the target resource. Users can edit all these elements, e.g., they associate the policy to another named graph, add or remove privileges, or modify the defined access conditions. Two different views are proposed to the user: i) a graphical view where operations are performed without the need to write policies using SPARQL and RDF to support *non-expert* administrators, and ii) a textual editor which allows to directly write policies using SPARQL and RDF for *expert* administrators.
- **Policies creation**: the creation of a new context-aware policy is managed by a wizard. In particular, the wizard proposes the following views: *i)* the definition of the policy name (which is then "translated" into an `rdfs:label`), the target named graph (it is possible to select one of the already defined named graphs included in the triple store, or to define a new one as we will detail later), and the privilege(s) to associate to the policy; *ii)* the view concerning the User dimension, that consists in a text box where the administrator inserts the features that must be satisfied by the user accessing the target resource, e.g., `foaf:knows :ACME_boss`. The text box provides autocompletion and it suggests a list of properties showing the associated vocabulary (to date, we use the `foaf`[3] and `relationship`[4] vocabularies, but other vocabularies can be added); *iii)* the view concerning the Environment dimension, that consists in two parts: the first one defines temporal conditions, and the second one deals with geographical conditions. Temporal conditions are expressed with a time picker, to select the desired time interval in which the access is granted. The definition of the geographical condition is done with a map interface[5], enriched with a movable marker and a resizable radius; *iv)* the view concerning the Device dimension, similar to the User view, that suggests the access properties related to the device used to access the target

[2] Video available at `http://wimmics.inria.fr/projects/shi3ld/`
[3] `http://xmlns.com/foaf/spec/`
[4] `http://purl.org/vocab/relationship/`
[5] `http://developers.google.com/maps/`

resource (we use the Delivery Context vocabulary[6] but further vocabularies can be added). At the end of the wizard, the access policy is automatically generated and stored in the triple store.

– **Named graphs creation**: the administrator is assisted in the definition of a new named graph. Shi3ld access policies must be associated to named graphs, and this leads to a number of difficult tasks for non-expert users, since it involves the use of non-trivial SPARQL features. We thus provide a GUI to mask such complexity, by letting administrators define a new named graph starting from the set of triples they want to associate to such newly defined named graph. The application asks for the label of the named graph to be created and it presents the template of a `SELECT` query, to be completed with the desired triple pattern. A preview of the selected triples is shown, thus letting the administrator check which triples will be added to the named graph. If results are satisfying, the new named graph is created and it can be used as the access policy target.

Figure 1 shows how user actions are translated into SPARQL and RDF by the Shi3ld Policy Manager. The application supports the administrator in creating, editing and deleting both policies and target named graphs. SPARQL queries are completely masked to end users, unless the embedded SPARQL textual editor is opened.

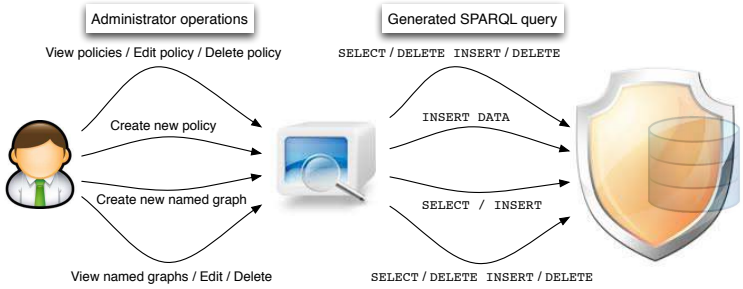

Fig. 1: The administrator operations and the resulting `SPARQL` query.

The Shi3ld Policy Manager is a web application developed in JavaScript and backed by a Fuseki SPARQL 1.1 triple store[7]. The server-side relies on the Node.js platform[8], and the front-end is built over jQuery, the Twitter Bootstrap framework[9], and Backbone.js[10] as structure. The SPARQL editor is provided by Flint[11].

[6] `http://www.w3.org/TR/dcontology/`
[7] `http://jena.apache.org/documentation/serving_data/`
[8] `http://nodejs.org/`
[9] `http://twitter.github.io/bootstrap/`
[10] `http://backbonejs.org/`
[11] `http://openuplabs.tso.co.uk/demos/sparqleditor`

3 Future Perspectives

We have presented a user interface to declare context-aware policies for the Shi3ld authorization framework. There are several issues to be considered as future research. First, since Shi3ld has been recently extended to manage also HTTP access to resources [3], we will extend this application such that also policies for Shi3ld-HTTP would be defined and manageable, i.e., access conditions are defined as RDF triples instead of `ASK` SPARQL queries. Second, we will integrate our interface with the Linked Open Vocabulary catalogue[12] such that administrators are supported in including new vocabularies used to define the access conditions. Third, we plan to favour policy reuse across datasets by adding a "policy template" sharing functionality. Moreover, we envision a "deep" properties validation, (i.e. checking that a certain URI actually corresponds to a `foaf` profile). Finally, we will add a sandbox to test the access policies effectiveness on the protected triples.

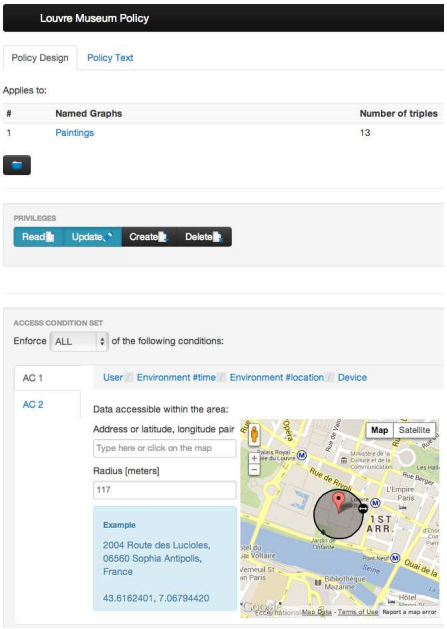

Fig. 2: The Shi3ld user interface.

References

1. Carroll, J.J., Bizer, C., Hayes, P.J., Stickler, P.: Named graphs. J. Web Sem. 3(4), 247–267 (2005)
2. Costabello, L., Villata, S., Gandon, F.: Context-Aware Access Control for RDF Graph Stores. In: Procs of ECAI. Frontiers in Artificial Intelligence and Applications, vol. 242, pp. 282–287. IOS Press (2012)
3. Costabello, L., Villata, S., Rocha, O.R., Gandon, F.: Access Control for HTTP Operations on Linked Data. In: Procs of ESWC. Lecture Notes in Computer Science, vol. 7882, pp. 185–199. Springer (2013)
4. Lopez, V., Uren, V.S., Sabou, M., Motta, E.: Is Question Answering fit for the Semantic Web?: A survey. Semantic Web 2(2), 125–155 (2011)
5. Ngomo, A.C.N., Bühmann, L., Unger, C., Lehmann, J., Gerber, D.: Sorry, i don't speak SPARQL: translating SPARQL queries into natural language. In: Procs of WWW. pp. 977–988. ACM (2013)
6. Sonntag, D., Heim, P.: A Constraint-Based Graph Visualization Architecture for Mobile Semantic Web Interfaces. In: Procs of SAMT. Lecture Notes in Computer Science, vol. 4816, pp. 158–171. Springer (2007)

[12] `http://lov.okfn.org/dataset/lov/`